

EDR Internals and Bypasses

Who Am I?

Consultant at Context
Information Security

@Kharosx0

Goals

- Demonstrate EDR capabilities
- Analyze Internal details of EDRs
- Demonstrate complete bypasses
- Live demo fun 😊

Contents

- EDR Introduction
 - General Capabilities
 - General Components
- Windows & EDR Internals
 - Drivers & Devices
 - Callbacks
 - Patchguard
 - Self-Protection
- Common Weaknesses + Live Demos

Why EDRs?

General Capabilities

- AV-Like behavior
- “Zero-day” protection
- Behavioral based detection
- Technical details? Not really
- Free trial?

REPLACE YOUR AV WITH **BETTER** PROTECTION

- ➔ Protects your endpoints against all threat types – known and unknown, malware and malware-free
- ➔ Combines machine learning malware protection, Indicator of Attack (IOA) behavioral blocking and exploit blocking for ultimate protection
- ➔ Eliminates reliance on signatures
- ➔ Requires no updates
- ➔ Delivers full protection

Predict and Prevent

Cyberattackers are innovating faster than traditional defenses can keep up. CB Defense uses advanced predictive models to analyze complete endpoint data and uncover malicious behavior to stop all types of attacks before they compromise your system.

- 1 Stop malware, ransomware, and non-malware attacks

The Cylance Predictive Advantage

SE Labs tested CylancePROTECT in an offline environment against major threats in the wild. Using our AI model from May 2015, SE Labs collected and tested threats from February 2016 to November 2017.

The results demonstrate CylancePROTECT users would have been safe from the zero-day attacks even if they had not updated their software for up to two years, nine months.

Prevention Capabilities

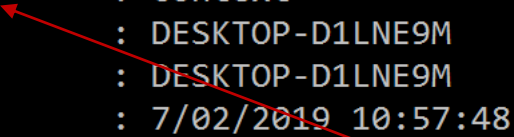
AntiVirus:

- Obfuscation modifications:

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonPasswords

Authentication Id : 0 ; 223780 (00000000:00000000)
Session           : Interactive from 1
User Name         : context
Domain            : DESKTOP-D1LNE9M
Logon Server      : DESKTOP-D1LNE9M
Logon Time        : 7/02/2019 10:57:48 PM
```



Works

EDR:

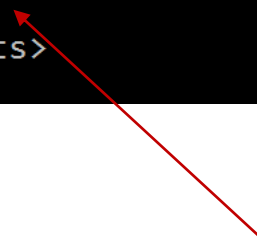
- Obfuscation modifications:

```
c:\Users\context\Documents>mimikatz_64.exe

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::logonPasswords

c:\Users\context\Documents>
```

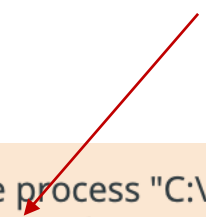


Terminated

Detection & Logging

- Powerful logging + querying = 😞

!?

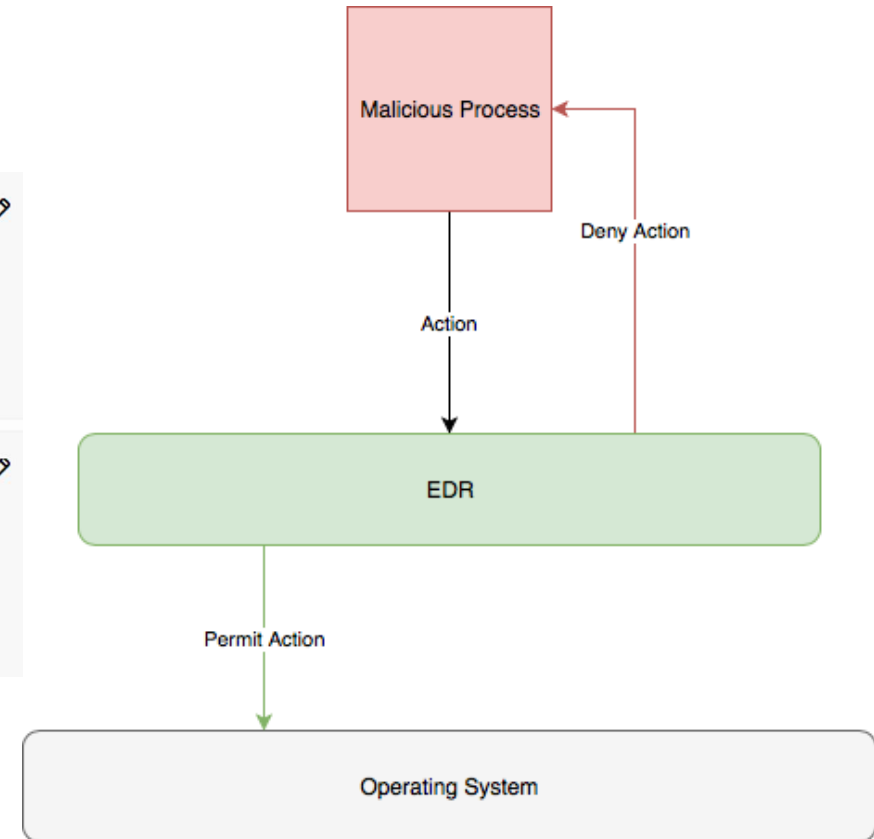


The application [C:\windows\system32\werfault.exe](#) attempted to open the process "C:\Program Files\VMware\VMware Tools\vmtoolsd.exe", by calling the function "OpenProcess". The operation was successful.

The application [C:\programdata\microsoft\windows defender\platform\4.18.1812.3-0\msmpeng.exe](#) established a TCP/443 connection to 40.115.119.185:443 (wdcp.microsoft.com, located in Dublin 07, Ireland) from 10.0.2.15:51144. The device was off the corporate network using the public address 5.147.241.30 (DESKTOP-D1LNE9M.contextis-testing.com, located in Dusseldorf 07, Germany). The operation was successful.

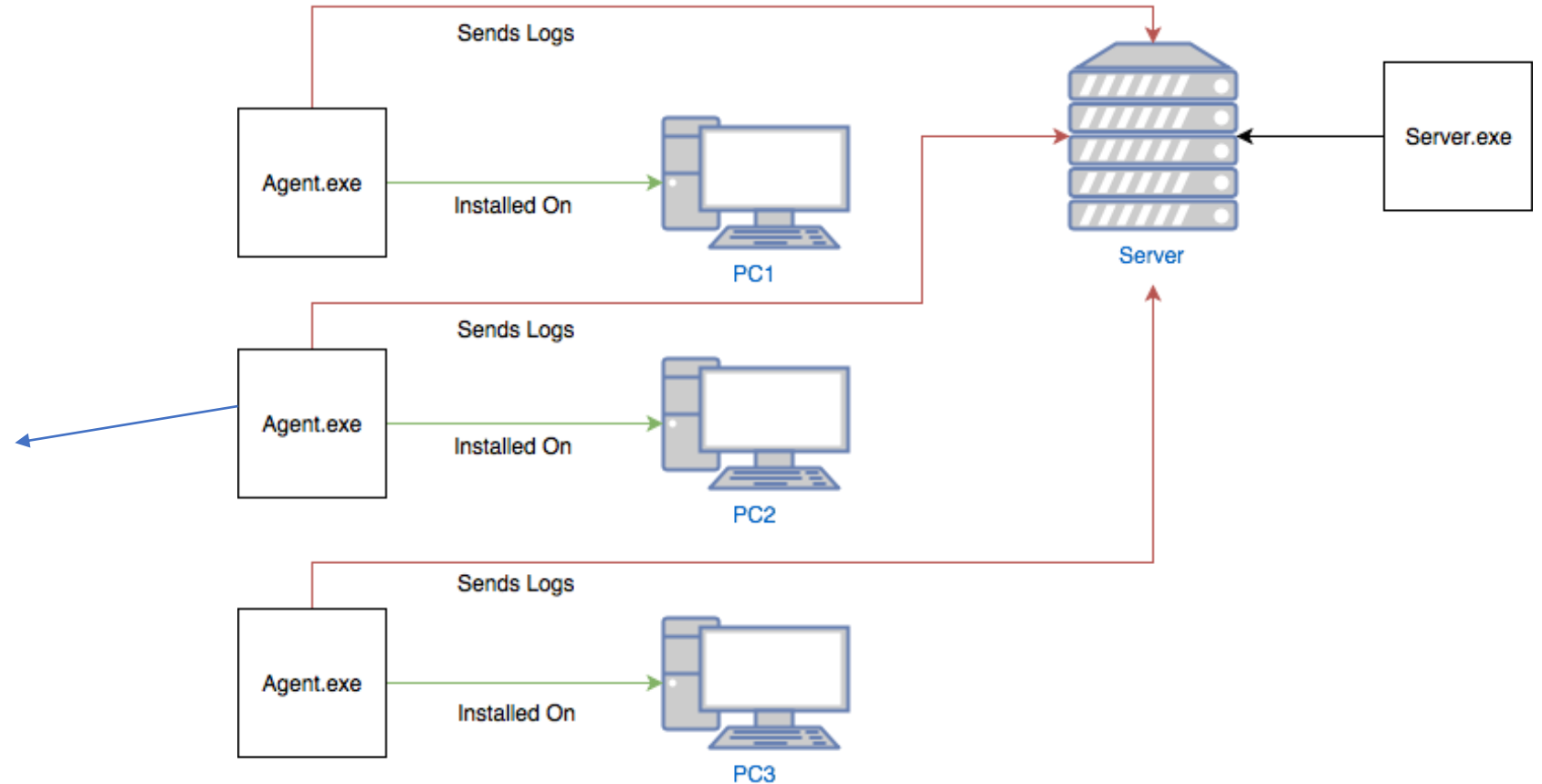
General Capabilities Pt.2

Not listed application	<div>Scrapes memory of another process</div> <div>Invokes a command interpreter</div> <div>Performs ransomware-like behavior</div> <div>Executes a fileless script</div> <div>Injects code or modifies memory of another process</div>	<div>Terminate process</div> <div>Terminate process</div> <div>Terminate process</div> <div>Terminate process</div> <div>Terminate process</div>
Application(s) at path: <div>**\javaw.exe, **\regsvr32.exe, **\rundll32.exe, **\mshta.exe, **\msbuild.exe, **\java.exe</div>	<div>Invokes a command interpreter</div> <div>Injects code or modifies memory of another process</div>	<div>Terminate process</div> <div>Terminate process</div>

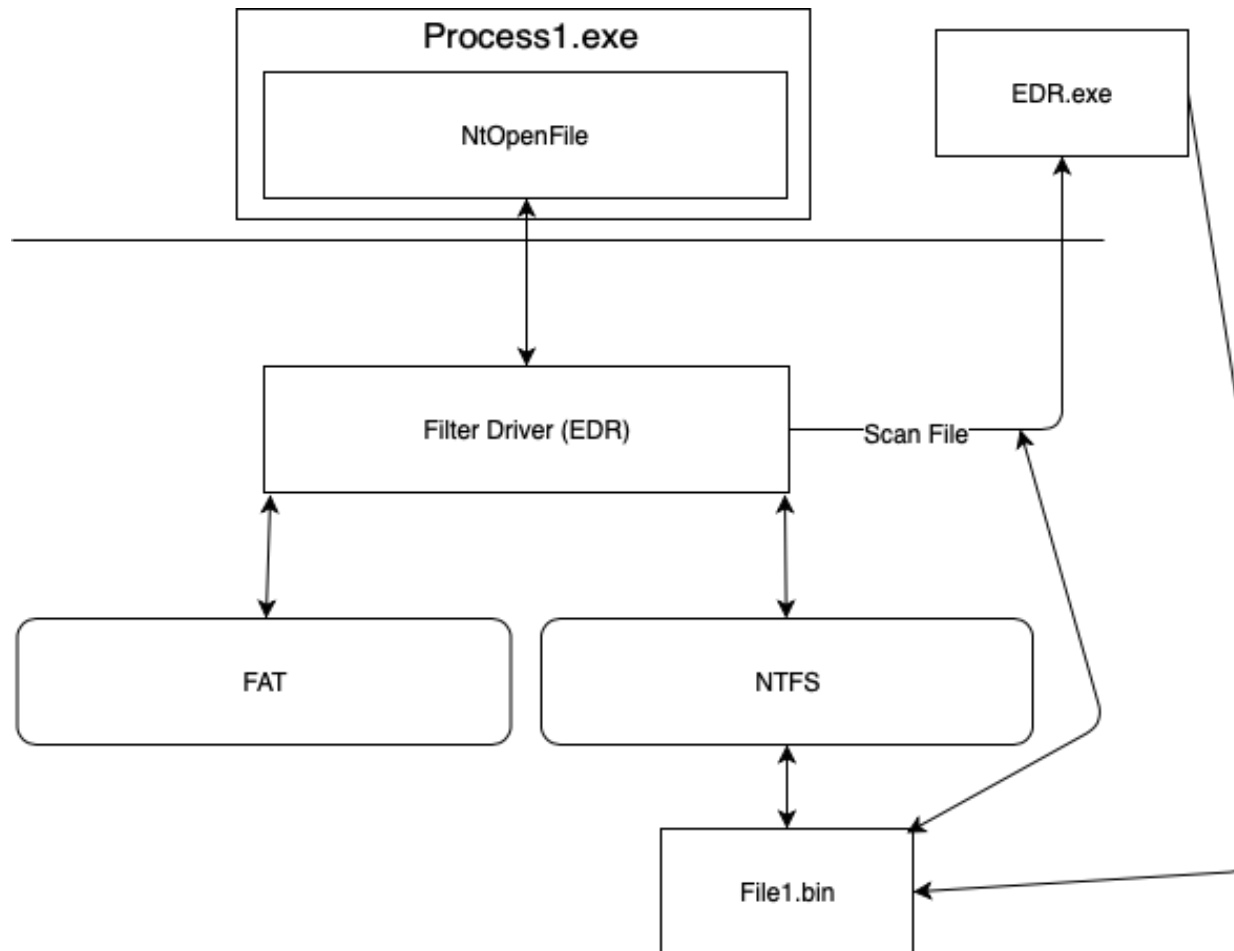


General Components

- Filter Drivers
- Network Drivers
- Software Drivers/Components
- Userland Apps



Key Component – filter Driver



Windows & EDR Internals

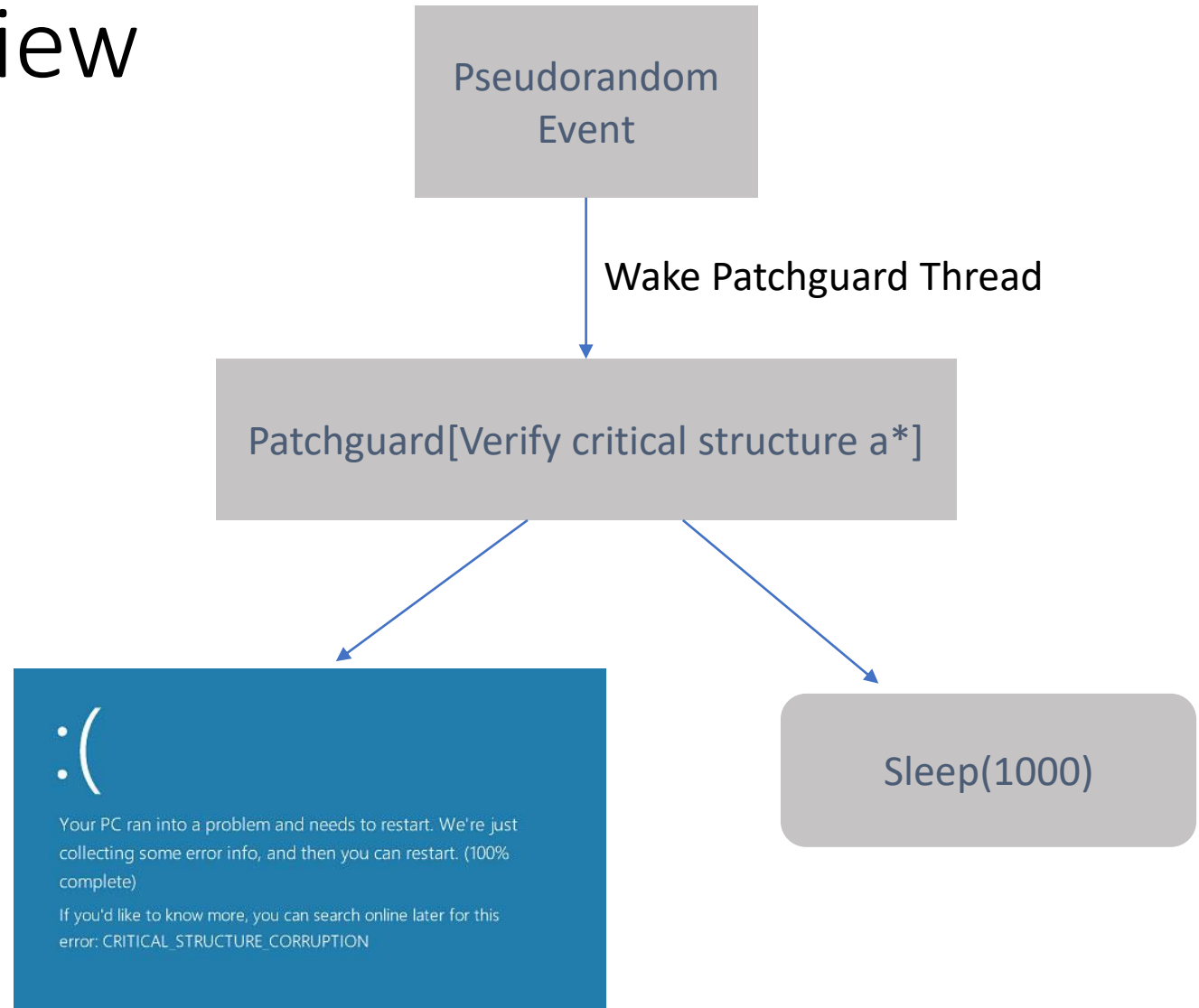
Callbacks

- PsSetCreateProcessNotifyRoutine
- PsSetLoadImageNotifyRoutine
- PsSetCreateThreadNotifyRoutine
- ObRegisterCallbacks (Handles)
 - PsProcessType
 - PsThreadType
- CmRegisterCallbacks (Registry)
- Demo: Viewing callbacks

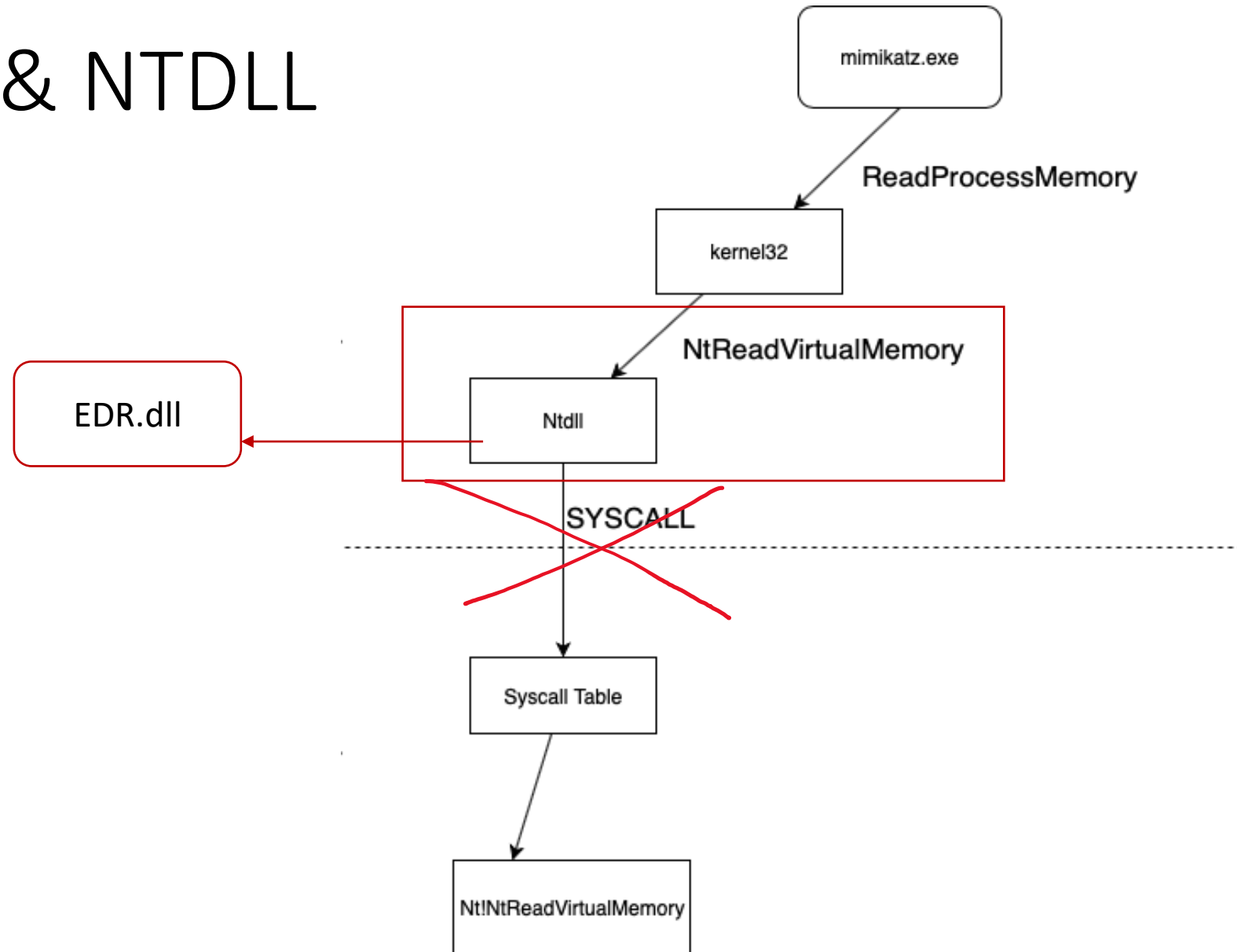
```
lea rcx, ProcessCallback
xor edx, edx
call cs:PsSetCreateProcessNotifyRoutineEx
```

Address	Ordinal	Name	Library
FFFFF80...		PsSetCreateProcessNotifyRoutineEx	ntoskrnl
FFFFF80...		PsSetLoadImageNotifyRoutine	ntoskrnl
FFFFF80...		PsRemoveLoadImageNotifyRoutine	ntoskrnl

Patchguard Overview



SYSCALLS & NTDLL



SYSCALLS pt.2

```
0:000> u ntdll!NtWriteVirtualMemory
ntdll!NtWriteVirtualMemory:
00007ffe`b2763bc0 4c8bd1          mov     r10,rcx
00007ffe`b2763bc3 b83a000000       mov     eax,3Ah
00007ffe`b2763bc8 0f05            syscall
00007ffe`b2763bca c3              ret
```

```
ntdll!NtWriteVirtualMemory:
*** ERROR: Symbol file could not be found.  Defaulted to export symbols
00007ffe`b2763bc0 e9cb6e5bf3       jmp     ctiuser!si_user_export+0x4f40
00007ffe`b2763bc5 0000            add     byte ptr [rax],al
00007ffe`b2763bc7 000f            add     byte ptr [rdi],cl
00007ffe`b2763bc9 05c30f1f44       add     eax,441F0FC3h
00007ffe`b2763bce 0000            add     byte ptr [rax],al
```

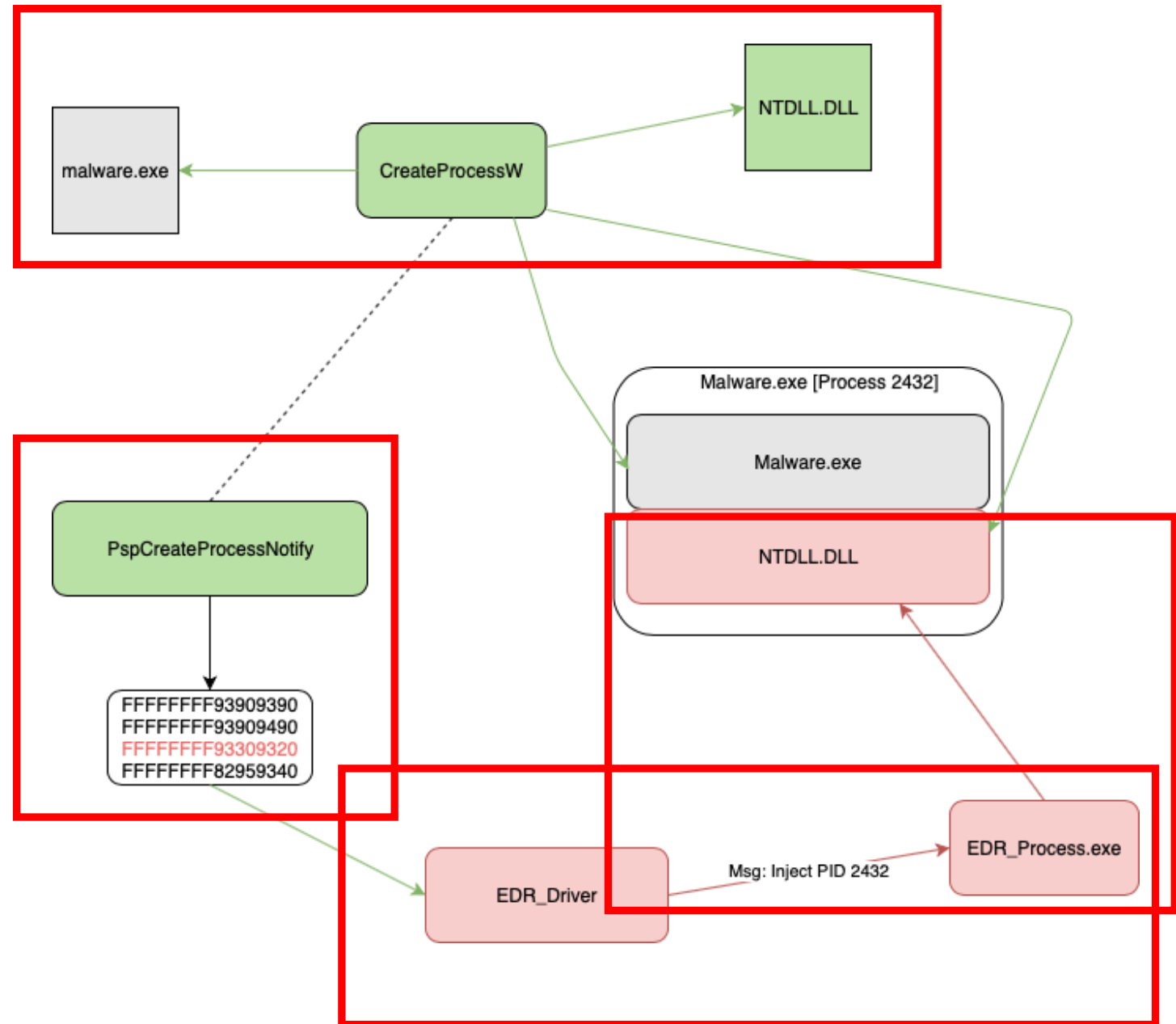
```
mov     r10,rcx
mov     eax,3Ah
jmp     ntdll!NtWriteVirtualMemory+0x8
```

```
0f05      syscall
c3        ret
```



Q2. How Does The Hooking Occur?

- Driver -> FltSendMessage (or similar)
- UserApp -> FltGetMessage
- Message = Inject Process x
- UserApp -> ProcessOpen & VirtualProtectEx & WriteProcessMemory



Self-Protection

- Common protections:
 - Uninstall code requirement (commonly optional)
 - Can't delete/modify installation folder/files
 - Prevent process termination
- Interestingly, the following protections weren't commonly found:
 - Obfuscation
 - Some EDRs were kind enough to contain debug messages
 - In-memory self integrity checks
 - Failed uninstall attempts logging

DbgPrints Everywhere

```
edx, esi  
[rsp+588] aLogprocessnoti: ; DATA XREF: sub_FFFF80009FC5354+63E1o  
sub_FFFF text "UTF-16LE  
r8d, word ptr [r15]  
rcx, aLogprocessno_0 ; "LogProcessNoti  
r8d, 4  
edx, esi  
DbgPrint  
_
```

```
; CHAR aReputationServ[]  
aReputationServ db '!!! Reputation Serv  
  
db 'g.',0Ah,0
```

```
; CHAR aHandlepidsetBa[]  
aHandlepidsetBa db 'handlePi  
  
db 'x. overwriteName: %u'.0Ah.0
```



```
get fullpath return NULL',0Ah  
FFFFF80009FD86C0:loc_FFFF80009FD895E1o
```

```
ss mode: %d',0Ah,0  
A XREF: sub_FFFF80009FCFEFC+AE1o
```

```
0[]  
!!! Reputation Service --- Message Response Format not recognized'  
; DATA XREF: sub_FFFF80009FD3124+CC1o  
vb a
```

Common Weaknesses

Uninstall.exe > EDR

- Uninstalling EDR from the machine is entirely viable
- Password = optional
- Secure passwords?
- Answer: Sometimes, Impossible
- Why?
 - Limited character set (alphanumeric (caps only))
 - Static, relatively small password length of 8
 - Bruteforceable?

XCP19QCQWGZ#HDXIGSJ

Install code

VS



Require code to uninstall sensor

Company Deregistration Code

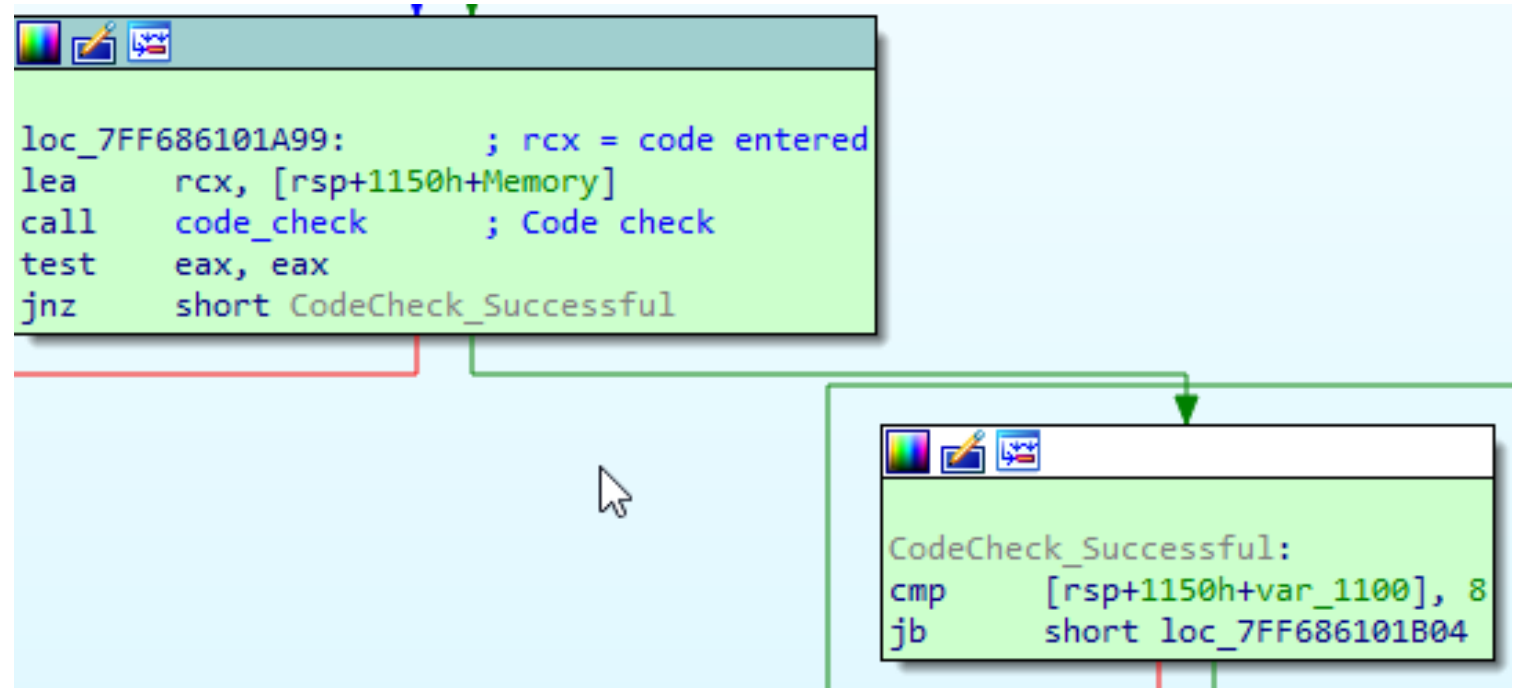
This is your company code which can be used for uninstalling sensors from endpoints if their policy requires it.

JURKR3BU

Uninstall code

Uninstall.exe Pt.2

- How about something easier, like falsifying the code check result?



For fun: Removing Callbacks

- Demo!

Bypassing Hooks Completely

- EDR's main behavioural detection capabilities come from a hooked NTDLL loaded into most processes, therefore a few bypasses exist:
 - Inline assembly
 - Dynamically loading an unhooked NTDLL
- Result? Complete bypass
- Demo!